

RESEARCH

Open Access



# THRIVE: threshold homomorphic encryption based secure and privacy preserving biometric verification system

Cagatay Karabat<sup>1\*</sup>, Mehmet Sabir Kiraz<sup>1</sup>, Hakan Erdogan<sup>2</sup> and ErKay Savas<sup>2</sup>

## Abstract

In this paper, we introduce a new biometric verification and template protection system which we call THRIVE. The system includes novel enrollment and authentication protocols based on threshold homomorphic encryption where a private key is shared between a user and a verifier. In the THRIVE system, only encrypted binary biometric templates are stored in a database and verification is performed via homomorphically randomized templates, thus, original templates are never revealed during authentication. Due to the underlying threshold homomorphic encryption scheme, a malicious database owner cannot perform full decryption on encrypted templates of the users in the database. In addition, security of the THRIVE system is enhanced using a two-factor authentication scheme involving user's private key and biometric data. Using simulation-based techniques, the proposed system is proven secure in the malicious model. The proposed system is suitable for applications where the user does not want to reveal her biometrics to the verifier in plain form, but needs to prove her identity by using biometrics. The system can be used with any biometric modality where a feature extraction method yields a fixed size binary template and a query template is verified when its Hamming distance to the database template is less than a threshold. The overall connection time for the proposed THRIVE system is estimated to be 336 ms on average for 256-bit biometric templates on a desktop PC running with quad core 3.2 GHz CPUs at 10 Mbit/s up/down link connection speed. Consequently, the proposed system can be efficiently used in real-life applications.

**Keywords:** Biometric; Security; Privacy; Cryptography; Homomorphic encryption; Malicious attacks

## 1 Introduction

In recent times, public and commercial organizations invest on secure electronic authentication (e-authentication) systems to reliably verify identity of individuals. Biometrics is one of the rapidly emerging technologies for e-authentication systems [1]. However, it is almost impossible to discuss biometrics without addressing the associated security and privacy concerns [2, 3]. Biometric data, stored either in a smart card or in a central database, incurs security and privacy risks due to increased number of attacks against identity management systems in recent years [2–5]. Security and privacy concerns on biometrics limit their widespread usage in real-life applications.

Biometric systems, which use error correction methods are proposed to cope with the noisy nature of the biometric templates in the literature [6–8]. These systems can obtain error-free biometric templates by using error correction techniques and thus cryptographic primitives (e.g., encryption or cryptographic hash) can successfully be employed free of the aforementioned avalanche effect [8–11]. However, their high error correcting capability requirements may render them impractical for real-life applications [12]. Furthermore, side information (e.g., parity bits) is needed for error correction and this may lead to information leakage (i.e., some attacks like error correcting code statistics, and non-randomness attacks) [13]. Zhou et al. successfully demonstrate that redundancy in an error correction code causes privacy leakage for biometric systems [14, 15].

\*Correspondence: cagatay.karabat@tubitak.gov.tr

<sup>1</sup>TÜBİTAK BİLGEM UEKAE, Gebze, Kocaeli, Turkey

Full list of author information is available at the end of the article

Although biometric template protection methods are proposed to overcome security and privacy issues [3, 16–29], recent research shows that security issues remain to be of a major concern [30–36]. In addition to this, there are a number of studies pointing out the privacy leakage of biometric applications [37, 38] as well as biometric template protection methods [14, 15, 39]. In the literature, Zhou et al. propose a framework for security and privacy assessment of the biometric template protection methods [14]. Also, Ignatenko et al. analyze the privacy leakage in terms of the mutual information between the public helper data and biometric features in a biometric template protection method. A trade-off between maximum secret key rate and privacy leakage is given in the works of Ignatenko et al. [38, 40].

Recently, homomorphic encryption methods are used with biometric feature extraction methods to perform verification via encrypted biometric templates [20, 41–43]. However, these methods offer solutions only in the semi-honest model where each party is obliged to follow the protocol but can arbitrarily analyze the knowledge that it learns during the execution of the protocol to obtain some additional information. The existing systems have not been designed for the malicious model where each party can arbitrarily deviate from the protocol and may be corrupted. They also do not take into account security and privacy issues of biometric templates stored in the database [20, 43]. The authors state that their security model will be improved in the future work by applying encryption methods also on the biometric templates stored in the database. Moreover, some of these systems are exclusively designed for a single biometric modality or a specific feature extraction method which also limits their application areas [41, 42]. In addition, an adversary can enroll herself on behalf of any user to their systems since they offer no protection against malicious enrollment. Finally, all of these systems suffer from computational complexity.

Biohashing schemes, one of the emerging biometric template protection methods [25–29], offer low error rates and fast verification at the authentication stage. However, they are vulnerable to attacks as reported in the literature [33–36]. These schemes should be improved to be safely adapted in a wide range of real-life applications. In this study, we present new enrollment and authentication protocols to increase the security and the privacy of the biometric verification. The proposed THRIVE system can work with any biometric feature extraction scheme whose templates are binary or can be binarized (e.g., by using thresholded random projection schemes) and where the verification decision can be based on Hamming distances between templates.

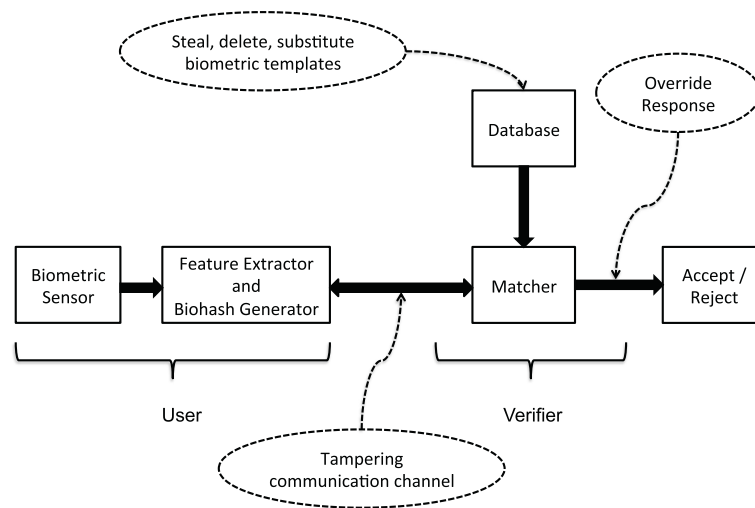
### 1.1 Problem definition

A simplified view of biometric authentication systems can be described as in Fig. 1. A user claims an identity and provides a biometric which is compared with existing biometric templates stored in a database and if the new biometric matches the one in the database, the user is verified to be a genuine user of the system. The database biometric templates are obtained during a separate enrollment session. There are security and privacy concerns related to different parts of this system as can be seen in Fig. 1. One of the main concerns is the protection of privacy in storage of database biometric templates to make sure that even if the templates are obtained by an adversary, they should not reveal any private information about the users of the system. Other concerns are eavesdropping or tampering with the communication channel between the user and the verifier or overriding the response of the system by adversaries.

A direct solution to biometric template protection might be to store biometric templates on the server side in plain form, and, in this way, it may seem to be possible to realize biometric authentication by simply utilizing well-known authentication protocols like SSL/TLS. However, as far as security and privacy are concerned, one of the major security issues in this scenario is that any malicious behavior on server side can be very harmful because of storage of biometric data in clear. Furthermore, these standard authentication protocols use conventional cryptographic primitives like Hash, RSA, and AES that cannot be directly used since encryptions cannot be decrypted by the server alone and also biometric data are inherently noisy [9]. More precisely, when a malicious database manager obtains decryption keys, she can perform decryption alone and can access biometric templates of all users. Therefore, biometric templates should be encrypted during the enrollment phase where the private key and encryption are never given to the server and authentication should be still guaranteed. Namely, when a biometric template is encrypted during the enrollment stage, only an approximate comparison between the stored and measured biometric data should be decrypted during the authentication stage. With the conventional cryptographic mechanisms, however, this again can lead to security and privacy issues for biometric templates at the authentication stage [9].

### 1.2 Our contributions

In this paper, we address adversary attacks in case of an active attacker who wants to gain unauthorized access to the system in the malicious attack model, where dishonest parties can deviate from the protocol and behave arbitrarily. By taking possible attacks into consideration, we propose a new biometric authentication system based on threshold homomorphic encryption. Our aim is to



**Fig. 1** Problem definition. A schematic description of a biometric authentication system and problems associated with various attacks

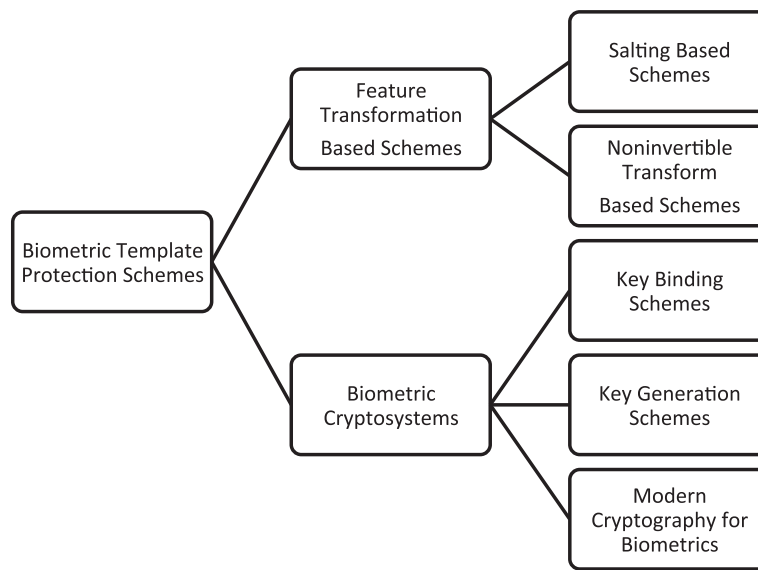
enhance the security of the system and preserve privacy of the users. The contributions of this work can be summarized as follows:

- A new biometric authentication system (which we call THRIVE) is proposed in the malicious model. It is a two-factor authentication system (biometric and secret key) and can be used in the applications where the user and the verifier do not trust each other.
- THRIVE system can be used with any existing biometric modality whose templates can be transformed into a binary vector and decision can be based on Hamming distances between binary templates.
- Even if an adversary gains access to the database and gathers encrypted biometric templates, she can neither authenticate herself by using these templates nor decrypt them due to the (2, 2)-threshold homomorphic encryption scheme.
- Biometric templates are never released even during the authentication phase and only the encrypted binary templates are stored in the database.
- The THRIVE system offers a new and advanced biometric template protection method without any helper data.
- The THRIVE system ensures security of the communication channel between the user and the verifier since all exchanged messages are randomized and/or encrypted. In addition, usage of nonces<sup>1</sup> and signature schemes guarantees uniqueness of the communication session.
- Since the biometric templates are encrypted, they are irreversible by definition as long as decryption keys, which are shared secrets, are not stolen.
- The THRIVE system can generate a number of protected templates from the same biometric data of a user because of randomized encryption. Thus, it ensures diversity. Besides, they are also cancelable, i.e., when they are stolen, they can be reproduced.
- The proposed authentication protocol run requires 336 ms and 671 ms on average for 256-bit and 512-bit biohash vectors, respectively, on a desktop PC with quad-core 3.2 GHz CPUs at 10 Mbit/s up/down link connection speed. Therefore, the THRIVE system is sufficiently efficient to be used in real-world applications.

The paper is structured as follows. Related work is presented in Section 2. Preliminaries are given in Section 3. The proposed biometric authentication system is introduced in Section 4. The security proofs of the proposed protocols are given in Section 5. The complexity analysis of the proposed system is discussed in Section 6. Comprehensive comparison between the proposed system and the existing systems/methods is given in Section 7. Finally, Section 8 concludes the paper.

## 2 Related work

In this part, we perform literature review on the works which are developed for mitigating security and privacy problems of biometrics and we categorize them as illustrated in Fig. 2. Jain et al. classify biometric template protection schemes into two main categories [3]: 1) Feature transformation-based schemes, and 2) Biometric cryptosystems. We analyze various studies under these two groups below.



**Fig. 2** Classification. Classification of biometric template protection methods

### 2.1 Feature transformation-based schemes

Feature transformation-based schemes use a transformation which is applied on a biometric template. The basic idea is to transform features of a biometric template into another secure domain by using a user defined secret key or password which determines parameters of the transformation.

Biohashing schemes are simple yet powerful biometric template protection methods [25–29] that can be classified under salting-based schemes. It is worth pointing out that biohashing is completely different from cryptographic hashing. Originally proposed to solve the aforementioned security and privacy issues, biohashing schemes fail to be a comprehensive solution. In [33–36], the authors claim that biohashes can be reversible under certain conditions and an adversary can estimate the biometric template of a user from her biohash. Consequently, when biohashes are stored in the databases and/or smart cards in their plain form, they can threaten the security of the system as well as the privacy of the users. Moreover, an adversary can use a compromised biohash to exploit the system security by performing malicious authentication. Also, as in the case of a compromised secret key, an adversary can recover the biometric template since these schemes are generally invertible [3].

Non-invertible transform-based schemes utilizing one-way functions have been proposed to protect biometric templates [44–46] against inversion attacks. A user secret key, which determines the parameters of non-invertible transformation function, is provided during the authentication stage. Even if an adversary obtains the secret key and/or the transformed biometric template, it is

computationally hard to recover the original biometric template. However, these schemes suffer from the trade-off between discriminability and non-invertibility which limits their recognition performance [3].

### 2.2 Biometric cryptosystems

The main idea behind biometric cryptosystems (also known as biometric encryption systems) is to use cryptographic techniques to enable template protection. Earlier methods include binding a cryptographic key with a biometric template or generating the cryptographic key directly from the biometric template [47]. Recently, there have been studies which use homomorphic encryption techniques for secure computation of Hamming distances. Thus, the biometric cryptosystems can be classified into three main categories: 1) Key binding schemes, 2) Key generation schemes, and 3) Modern cryptography for biometrics.

Key binding and generation systems use helper data, which is public information, about the biometric template for verification. Although helper data is supposed to leak no critical information about the biometric template, Rathgeb et al. show that helper data is vulnerable to statistical attacks [48]. Furthermore, Ignatenko et al. show how to compute a bound on possible secret rate and privacy leakage rate for helper data schemes [49]. Adler conducts a hill-climbing attack against biometric encryption systems [31]. In addition, Stoianov et al. propose several attacks (i.e., nearest impostors, error correcting code statistics, and non-randomness attacks) to biometric encryption systems [13].

### 2.2.1 Key binding systems

In the literature, fuzzy commitment [8] and fuzzy vault schemes [24] are categorized under the key binding schemes. These schemes aim to bind a cryptographic key with a biometric template and it is expected that neither the biometric template nor the random bit string can be recovered if the corresponding user's biometric data is not known. However, this is not the case in reality as biometric templates are not uniformly random. Furthermore, error correction codes (ECC) used in biometric cryptosystems lead to statistical attacks (i.e., running ECC in a soft decoding or erasure mode and ECC Histogram attack) [13, 50]. Ignatenko et al. show that fuzzy commitment schemes leak information in cryptographic keys and biometric templates which lead to security flaws and privacy concerns [38, 40]. In addition, Zhou et al. argue that fuzzy commitment schemes leak private data. Chang et al. describe a non-randomness attack against fuzzy vault scheme which causes distinction between the minutiae points and the chaff points [51]. Moreover, Kholmatov et al. describe a correlation attack against fuzzy vault schemes [52].

### 2.2.2 Key generation systems

Keys are generated from helper data and a given biometric template in key generation schemes [3]. Fuzzy key extraction schemes are classified under the key generation ones and they use helper data [53–58]. These schemes can be used as an authentication mechanism where a user is verified via her own biometric template as a key. Although the fuzzy key extraction schemes produce keys from biometric templates, the repeatability and the randomness of the generated keys are two major questions [3]. Boyen et al. describe several vulnerabilities of the fuzzy key extraction schemes from an attacker perspective [59], e.g., improper fuzzy sketch constructions leading to information leakage on the secret, biased codes allowing majority vote attack, and permutation leaks. Moreover, Li et al. argue that when an adversary obtains sketches, they may reveal the identity of the users [60].

### 2.2.3 Modern cryptography for biometrics

In recent years, a number of papers have been published on systems in which biometrics and homomorphic encryption work together for either authentication or identification purposes. These systems have cryptographic protocols based on secure multiparty computation and most of them especially use superior properties of homomorphic encryption schemes (e.g., allow computation on encrypted data) in order to overcome security and privacy threats to the biometric data.

Kerschbaum et al. [61] propose a protocol to compare fingerprint templates without actually exchanging them by using secure multi-party computation in the

semi-honest model. At the enrollment stage, the user gives her fingerprint template, minutiae pairs and a PIN to the system. Thus, the verifier knows the fingerprint templates which are collected at the enrollment stage. Although the user does not send her biometric data at the authentication, the verifier already has the user's enrolled biometric data and this violates the privacy of the user in case of a malicious (or compromised) verifier. A malicious verifier can use these fingerprint templates for malicious authentication. Furthermore, since the fingerprint comparison reveals the matching scores (e.g., Hamming distance [62]), the attacker can launch a hill climbing attack against this system.

Erkin et al. [41] propose a privacy preserving face recognition system for the eigen-face recognition algorithm [63]. They present a protocol that performs operations on encrypted images by using the Paillier homomorphic encryption scheme. Later, Sadeghi et al. improve the efficiency of this system [42]. In both studies, the system is limited to using eigen-face recognition algorithm with homomorphic encryption when there are better alternative face recognition algorithms. Moreover, they do not use a threshold cryptosystem, which prevents a malicious party from aiming to perform decryption by himself. Storing face images (or corresponding feature vectors) in the database in plain form is the most serious security drawback of this system. An adversary, who has access to the database, may obtain all face images.

Barni et al. [20, 43] propose a privacy preservation system for fingercode templates by using homomorphic encryption in the semi-honest model where all parties follow the protocol, but dishonest parties may be curious to violate others' privacy. Thus, they do not propose any security and privacy solutions on the biometric templates stored in the database. This issue is mentioned as a future work in their paper. In addition, they do not use threshold encryption, which would protect the system against a malicious party aiming to perform decryption by herself. Another drawback is that the system does not address the malicious enrollment issue. Although they achieve better performance than those in [41, 42] in terms of bandwidth and time complexity, they do not consider the scenarios where the user and the verifier do not trust each other (i.e., the malicious model).

Kulkarni et al. [64] propose a biometric authentication system based on *somewhat* homomorphic encryption scheme of Boneh et al. [65], which allows an arbitrary number of addition of ciphertexts, but supports only one multiplication operation between the ciphertexts. Although the values stored on the enrollment server are the XORed values of the biometric template vector with the corresponding user's key, the user first extracts and sends her biometric features to the trusted enrollment server. Again this system uses a trusted enrollment server

and fails to provide security and privacy objectives against a malicious database manager. In addition, the system is not efficient since 58 s is required for a successful authentication of a 2048-bit binary feature vector.

There are also some works on secure Hamming distance calculation by using cryptographic primitives [64, 66–68]. These papers, however, limit the scope of their works only to secure Hamming distance calculation. They do not address biometric authentication as a whole and do not satisfy security and privacy concerns under a malicious model.

Osadchy et al. [66] propose a secure Hamming distance calculation scheme based on Pailler homomorphic encryption for face biometrics. The system is called SCiFI. Although they claim that SCiFI is computationally efficient, it mostly uses pre-computation techniques. Its pre-computation time includes processing time that must be done locally by each user before using the system each time. They report that SCiFI's online running time takes 0.31 s for a face vector of size 900 bits; however, its offline computation time takes 213 s. SCiFI provides security against only in the semi-honest adversaries.

Rane et al. [67] also propose secure Hamming distance calculation for biometric applications. Nonetheless, their proposed method fails to ensure biometric database security because biometric templates are stored in plain format in the database. Thus, a malicious verifier can threaten a user's security and privacy. Bringer et al. [68] propose a secure Hamming distance calculation for biometric application, called as SHADE. This technique is based on committed oblivious transfer [69]. However, it cannot guarantee biometric database security since biometric templates are stored in plain form in the database.

### 3 Preliminaries

#### 3.1 Threshold homomorphic cryptosystem

In this section, we briefly describe underlying cryptographic primitives of the protocols. Given a public key encryption scheme, let  $m \in \mathcal{M}$  denote the message or plaintext space,  $c \in \mathcal{C}$  the ciphertext space, and  $r \in \mathcal{R}$  its randomness. Let  $c = \text{Enc}_{pk}(m; r)$  depict an encryption of  $m$  under the public key  $pk$  where  $r$  is a random value. Let  $sk$  be the corresponding private key, which allows the holder to retrieve a message from a ciphertext. The decryption procedure is performed with the private key  $sk$  as  $m = \text{Dec}_{sk}(c)$ .

In a  $(t, n)$ -threshold cryptosystem, the knowledge of a private key is distributed among parties  $P_1, \dots, P_n$ . Then, at least  $t$  of these parties are required for a successful decryption. On the other hand, there is a public key to perform encryption. More formally, let  $P_1, \dots, P_n$  be the participants. We define a  $(t, n)$ -threshold encryption scheme with three phases as follows:

- In the *key generation* phase, each participant  $P_i$  receives a pair of  $(pk_i, sk_i)$ , where  $pk_i$  and  $sk_i$  are the shares of the public and secret key, respectively. Then, the overall public key  $pk$  is constructed by collaboratively combining the shares. Finally  $pk$  is broadcasted to allow anyone to encrypt messages in  $\mathcal{M}$ . The shares of this public key are also broadcasted to allow all parties to check the correctness of the decryption process.
- The *encryption* phase is performed as in any public key encryption cryptosystem. If  $m \in \mathcal{M}$  is the message, a (secret) random value  $r$  from  $\mathcal{R}$  is chosen and  $c = \text{Enc}_{pk}(m; r)$  is computed under a public key  $pk$ .
- In the *threshold decryption* phase, given that  $t$  (or more) participants agree to decrypt a ciphertext  $c$ , they follow two steps. First, each participant produces a decryption share by performing  $S_i^j = \text{Dec}_{sk_i^j}(c)$ ,  $j = 1, \dots, t$ . After broadcasting  $S_i^j$ , they all can apply a reconstruction function  $\mathcal{F}$  on these shares so that they can recover the original message by performing  $m = \mathcal{F}(S_1^1, \dots, S_t^t)$  where  $P_1^1, \dots, P_t^t$  represent the group of  $t$  participants willing to recover  $m$ .

In case of a  $(t, n)$ -threshold scheme, the additional requirement is that if less than  $t$  parties gather their correct shares of the decryption of a given ciphertext, they will gather no information whatsoever about the plaintext. In the proposed system, we use the  $(2, 2)$ -threshold cryptosystem between the prover (the user) and the verifier where both players must cooperate to decrypt. In this way, we ensure that the verifier cannot decrypt the ciphertexts alone, and the decryption is only performed by both the user and the verifier during the computation of Hamming distance of their corresponding inputs.

A public key encryption scheme is said to be additively homomorphic if given  $c_1 = \text{Enc}(m_1; r_1)$  and  $c_2 = \text{Enc}(m_2; r_2)$  it follows that  $c_1 c_2 = \text{Enc}(m_1 + m_2; r_3)$  where  $m_1, m_2 \in \mathcal{M}$  and  $r_1, r_2, r_3 \in \mathcal{R}$ . That is to say, homomorphic encryption is a form of encryption that allows parties to perform computations on the encrypted values and match the result of operations performed on the plaintexts. Namely, they do not possess the decryption key, and therefore they do not know the plaintexts but can still perform operations under encryption. Conventional cryptosystems do not satisfy the homomorphic property.

There are various versions of threshold homomorphic cryptosystems. The most widely used are ElGamal [70] or Paillier [71] cryptosystems. In our proposal, we will use a threshold version of Goldwasser-Micali (GM) encryption scheme (i.e., between a user and a verifier) proposed by Katz and Yung in [72]. GM scheme is XOR-homomorphic [73], i.e., given any two bits  $b_1, b_2$

in  $\{0, 1\}$ , any random values  $r_1, r_2 \in \mathcal{R}$ , and any encryptions  $\text{Enc}(b_1, r_1)$ ,  $\text{Enc}(b_2, r_2)$ , it is easy to compute  $\text{Enc}(b_1 \oplus b_2, r_1 r_2)$ . Note that our scheme requires encryption of bits instead of bit-strings, and GM encryption is more efficient compared to ElGamal or Paillier in the case of bit encryption.

In the proposed protocol, we use a variant of the threshold decryption protocol which is the so-called private threshold decryption [74]. The requirement of this protocol is that one of the  $t$  parties will be the only party who will recover the secret. All  $t - 1$  other parties follow the protocol and broadcast their shares to achieve this requirement. The party who will learn the plaintext proceeds with the decryption process privately, collects all decryption shares from the  $t - 1$  other parties, and privately reconstructs the message. The remaining parties will not get any information about the message.

### 3.1.1 Threshold XOR-Homomorphic Goldwasser-Micali encryption scheme

We next give a brief explanation of (2,2)-GM cryptosystem between two users (in our proposal, between a user and a verifier) using a Trusted Dealer. We note that one can also exclude a trusted dealer using the scheme in [72]:

#### Key generation:

The trusted dealer first chooses prime numbers  $p$  and  $q$  ( $\|p\| = \|q\| = n$ ) such that  $N = pq$  and  $p \equiv q \equiv 3 \pmod{4}$ . The dealer next chooses  $p_1, q_1, p_2, q_2 \in_R (0, 2^{2n})$  such that  $p_1 \equiv q_1 \equiv 0 \pmod{4}$  and  $p_2 \equiv q_2 \equiv 0 \pmod{4}$ . He sets  $p_0 = p - p_1 - p_2$  and  $q_0 = q - q_1 - q_2$  and sends  $(p_1, q_1)$  to the first party and  $(p_2, q_2)$  to the second party. He finally broadcasts  $(p_0, q_0, N)$ .

#### Encryption of a bit $b \in \{0, 1\}$ :

Choose  $r \in_R \mathbb{Z}_N$  and compute a ciphertext  $C = (-1)^b r^2 \pmod{N}$ .

#### Decryption:

All parties compute the Jacobi symbol  $J = \left(\frac{C}{N}\right)$ . If  $J \neq 1$  then all parties stop because either the encryption algorithm was not run honestly or the ciphertext was corrupted during the transmission.

(Note that  $\left(\frac{C}{N}\right)$  is always 1, because

$\left(\frac{C}{N}\right) = \left(\frac{C}{p}\right) \left(\frac{C}{q}\right) = 1$  (i.e., either  $\left(\frac{C}{p}\right) = 1$  and  $\left(\frac{C}{q}\right) = 1$  or  $\left(\frac{C}{p}\right) = -1$  and  $\left(\frac{C}{q}\right) = -1$ ). If  $J = 1$

then the first party broadcasts  $b_1 = C^{(-p_1 - q_1)/4} \pmod{N}$ . The second party (who is going to decrypt) will privately compute  $b_0 = C^{(N - p_0 - q_0 + 1)/4} \pmod{N}$  and  $b_2 = C^{(-p_2 - q_2)/4} \pmod{N}$ . Finally, the decrypted bit  $b$  is computed as  $b = (1 - b_0 b_1 b_2 \pmod{N}) / 2$ .

Note that it is easy to see whether  $C$  is a quadratic residue by computing  $b \equiv C^{(N - p - q + 1)/4} \pmod{N}$ . The reason is briefly as follows. We first note that by Euler's theorem  $C^{\phi(N)} \equiv 1 \pmod{N}$  where  $\phi(N) = (p - 1)(q - 1)$ . We also know that  $C$  is quadratic residue if and only if  $C^{\phi(N)/2} \equiv 1 \pmod{N}$ . If the Jacobi symbol  $J = \left(\frac{C}{N}\right) = 1$  then by using  $\left(\frac{C}{p}\right) \left(\frac{C}{q}\right) = 1$ , we have either  $\left(\frac{C}{p}\right) = 1$  and  $\left(\frac{C}{q}\right) = 1$  or  $\left(\frac{C}{p}\right) = -1$  and  $\left(\frac{C}{q}\right) = -1$ . If  $\left(\frac{C}{p}\right) = 1$  (resp.  $-1$ ) and  $\left(\frac{C}{q}\right) = 1$  (resp.  $-1$ ) then  $C^{p-1/2} \equiv 1 \pmod{p}$  (resp.  $-1 \pmod{p}$ ) and  $C^{q-1/2} \equiv 1 \pmod{q}$  (resp.  $-1 \pmod{q}$ ). Hence, for both cases  $C^{(p-1)(q-1)/4} \equiv 1 \pmod{p}$  and  $C^{(p-1)(q-1)/4} \equiv 1 \pmod{q}$ . By the Chinese Remainder Theorem, we have  $C^{(p-1)(q-1)/4} \equiv 1 \pmod{N}$ . Hence,  $C$  is quadratic residue if and only if  $b = 1$ .

### 3.2 Biometric verification scheme

Biometric verification schemes perform an automatic verification of a user based on her specific biometric data. They have two main stages: 1) Enrollment stage, and 2) Authentication stage. The user gives her biometric data to the system at the enrollment stage. Then in the authentication stage, she provides her biometric data to the system to prove her identity. Any biometric scheme, which provides binary templates or whose templates can be binarized, can be used with the proposed threshold homomorphic cryptosystem. Most biometric templates are represented as fixed length real vectors and they can be binarized easily using locality sensitive hashing (LSH) techniques and the most natural and widely used distance in the hash space is the Hamming distance. Previous studies have shown that coming up with distance preserving binary hashing is possible [75–77]. So, we believe the system we propose can be used in a broad class of verification systems with minor modification in the system to binarize the templates and use Hamming distance for distance calculation, which will result in minimal loss of security properties in the system (such as equal error rate (EER) etc.). In this paper, we use biohashing as an example algorithm for extracting binary biometric templates. Random projection and thresholding used in biohashing is a well-known type of LSH approach [78]. Although biohashing has its own security and privacy preservation mechanism, we do not rely on these to address the security or privacy concerns.

Biohash is a binary and pseudo-random representation of a biometric template. Biohashing schemes use two inputs: 1) Biometric template, 2) User's secret key. A biometric feature vector is transformed into a lower dimension sub-space using a pseudo-random set of orthogonal vectors which are generated from the user's secret key. Then, the result is binarized to produce a pseudo-random

bit-string which is called the biohash. In an ideal case, the Hamming distance between the biohashes belonging to the biometric templates of the same user is expected to be relatively small. On the other hand, the distance between the biohashes of different users is expected to be sufficiently high to achieve higher recognition rates.

We adopt the random projection (RP)-based biohashing scheme proposed by Ngo et al. [79]. In this scheme, there are three main steps: 1) Feature extraction, 2) Random projection, 3) Quantization. These steps are explained in the following.

### 3.2.1 Feature extraction

The feature extraction is performed on the biometric data (e.g., face image) which is collected at the enrollment stage, belonging to users,  $\mathbf{I}_{i,j} \in \mathbb{R}^{m \times n}$  for  $i = 1, \dots, n$  and  $j = 1, \dots, L$  where  $n$  and  $L$  denote the number of users and the number of training images per user, respectively. The images are lexicographically re-ordered and the training vectors,  $\mathbf{x}_{i,j} \in \mathbb{R}^{(mn) \times 1}$ , are obtained. Then, Principle Component Analysis (PCA) [63] is applied to these vectors as follows

$$\mathbf{y}_{i,j} = \mathbf{A}(\mathbf{x}_{i,j} - \mathbf{w}), \quad (1)$$

where  $\mathbf{A} \in \mathbb{R}^{k \times (mn)}$  is the PCA matrix trained by the images in the training set,  $\mathbf{w}$  is the mean face vector, and  $\mathbf{y}_{i,j} \in \mathbb{R}^{k \times 1}$  is the vector containing the PCA coefficients belonging to the  $j$ th training image of the  $i$ th user.

### 3.2.2 Random projection (RP)

An RP matrix,  $\mathbf{R} \in \mathbb{R}^{\ell \times k}$ , is generated to reduce the dimension of the PCA coefficient vectors. The RP matrix elements are independent and identically distributed (*i.i.d*) and generated from a Gauss distribution with zero mean and unit variance by using a Random Number Generator (RNG) with a seed derived from the user's secret key. The Gram-Schmidt (GS) procedure is applied to obtain an orthonormal projection matrix  $\mathbf{R}_{GS} \in \mathbb{R}^{\ell \times k}$  to have more distinct projections. Finally, the PCA coefficients are projected onto a lower  $\ell$ -dimensional subspace using

$$\mathbf{z}_{i,j} = \mathbf{R}_{GS}\mathbf{y}_{i,j} \quad (2)$$

where  $\mathbf{z}_{i,j} \in \mathbb{R}^{\ell \times 1}$  is an intermediate biohash vector belonging to the  $j$ th training image of the  $i$ th user.

### 3.2.3 Quantization

The elements of the intermediate biohash vector  $\mathbf{z}_{i,j}$  are binarized with respect to a threshold as follows

$$\lambda_{i,j}^k = \begin{cases} 1 & \text{if } z_{i,j}^k \geq \beta \\ 0 & \text{Otherwise,} \end{cases} \quad (3)$$

where  $\lambda_{i,j} \in \{0, 1\}^\ell$  denotes biohash vector of the  $j$ th training image of the  $i$ th user and  $\beta$  denotes the mean value of the intermediate biohash vector  $\mathbf{z}_{i,j}$ .

A biohash vector,  $\mathbf{B}_{enroll_i}$  for the  $i$ th user, which can be any vector among  $\lambda_{i,j}$  vectors in a real-world application, is stored in the database during the enrollment stage, which is accessed for verification purpose later during the authentication phase. For simulation purposes, we take into account all possible biohashes for a user by computing  $\lambda_{i,j}$ . The user is authenticated when the Hamming distance between  $\mathbf{B}_{enroll_i}$  and  $\mathbf{B}_{auth_i}$  is below a threshold  $\mu$ , where  $\mathbf{B}_{auth_i}$  is the biohash vector measured during the authentication stage as follows

$$\sum_{k=1}^n B_{enroll_i}^k \oplus B_{auth_i}^k \leq \mu, \quad (4)$$

where  $B_{enroll_i}^k$  denotes the  $k$ th bit of  $\mathbf{B}_{enroll_i}$ ,  $B_{auth_i}^k$  denotes the  $k$ th bit of  $\mathbf{B}_{auth_i}$ , and  $\oplus$  denotes the binary XOR (exclusive OR) operator. Consequently, the verifier decides whether the prover is an authorized user depending on the threshold,  $\mu$ .

## 4 The proposed biometric authentication system

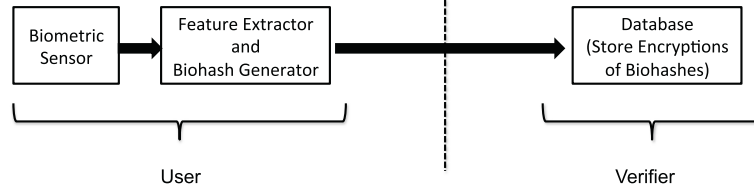
**stop** In this section, we introduce the THRIVE system which has two participants: *User* ( $U_i$ ) and *Verifier* (V). The user has control of the biometric sensor, the feature extractor, and the biohash generator whereas the verifier has control of the database and the matcher as can be observed in Fig. 3 in the enrollment stage and Fig. 4 in the authentication stage. We assume that there is a trusted third party (TTP) which initially sets up the system public/private keys.

The TTP distributes the keys in the proposed system. There are public-private key pairs  $(pk_i, (sk_i^1, sk_i^2))$ , which are shared between the user and the verifier. Here,  $sk_i^1$  is the private key share of the  $i$ th user,  $U_i$ , and  $sk_i^2$  is the private key share of the verifier.  $pk_i$  is the public key of the  $i$ th user,  $U_i$ , and both the user and the verifier know it. When an enrollment biometric template is encrypted by  $pk_i$ , this can solely be decrypted using the private key shares of the user ( $sk_i^1$ ) and the verifier ( $sk_i^2$ ) collaboratively since the proposed system is based on the (2, 2)-threshold homomorphic cryptosystem. In addition, there is another public-private key pair  $(pk_{U_i}, sk_{U_i})$ , which belongs to the  $i$ th user,  $U_i$ , where  $pk_{U_i}$  is the public key and  $sk_{U_i}$  is its associated private key to perform the signature operation. The verifier also knows the public key  $pk_{U_i}$ .

### 4.1 Enrollment stage

The proposed enrollment protocol is illustrated in Fig. 5 and its steps are introduced as follows:





**Fig. 3** Enrollment. Illustration of the THRIVE enrollment stage

1. **Step 1:** The  $i$ th user,  $U_i$ , computes her biohash,  $B_{enroll_i} = B_{enroll_i}^1 \cdots B_{enroll_i}^n$  where  $B_{enroll_i}^j \in \{0, 1\}$ ,  $j = 1, \dots, n$ . Next, the user encrypts her biohash,  $C_i^j = \text{Enc}_{pk_i}(B_{enroll_i}^j)$  for  $j = 1, \dots, n$ , by using the public key  $pk_i$ . Then, the user signs her encrypted biohash,  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ , and sends it to the verifier.
2. **Step 2:** The verifier  $V$  verifies  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$  by using  $pk_{U_i}$  and stores the signature and encrypted biohash in the database. This data will be used for verification at the authentication stage.

Note that both the user and the verifier have to cooperate to decrypt a ciphertext due to the  $(2, 2)$ -threshold homomorphic cryptosystem. Furthermore, the signature ensures that the data stored in the database is generated by an authorized user.

**Lemma 1.** *Biohashes are not revealed at the enrollment stage.*

*Proof.* (Sketch) At the enrollment stage, the  $i$ th user  $U_i$  first encrypts her biohash and then signs it. After these computations,  $U_i$  sends her encrypted and signed biohash  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$  to the verifier. Since

user's biohash is not sent in plain form, biohashes are not revealed to the verifier at the enrollment stage.  $\square$

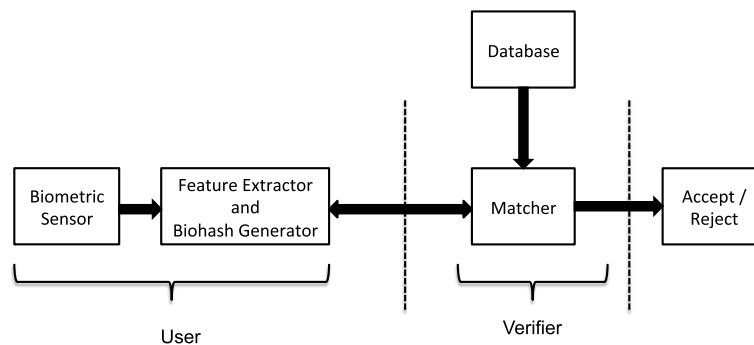
**Lemma 2.** *An adversary cannot register as a user at the enrollment stage.*

*Proof.* (Sketch) At the enrollment stage, the  $i$ th user  $U_i$  encrypts her biohash by using the public key  $pk_i$  and then signs her encrypted biohash by using her private key  $sk_{U_i}$ . Thus,  $U_i$  sends encrypted and signed biohash  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$  to the verifier. Since the verifier verifies the signature of the user, an adversary cannot register himself as a user without knowing the user private key  $sk_{U_i}$  used to compute  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ .  $\square$

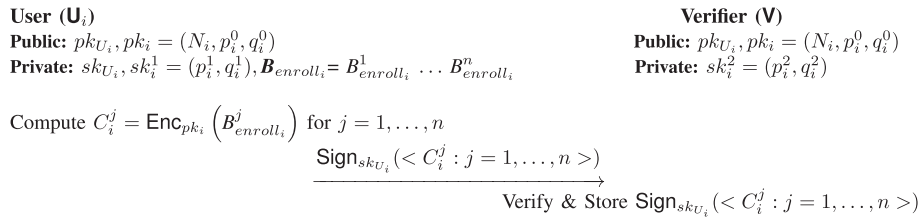
#### 4.2 Authentication stage

In this stage,  $U_i$  tries to prove herself to the verifier by executing the proposed authentication protocol shown in Fig. 6. Similar to the enrollment case, the biometric sensor must be authorized by the system before the authentication protocol is carried out. Steps of the protocol are given as follows:

1. **Step 1:**  $U_i$  wants to verify her identity by using her biohash and sends a connection request to the verifier. Then,  $U_i$  computes her biohash  $B_{auth_i} =$



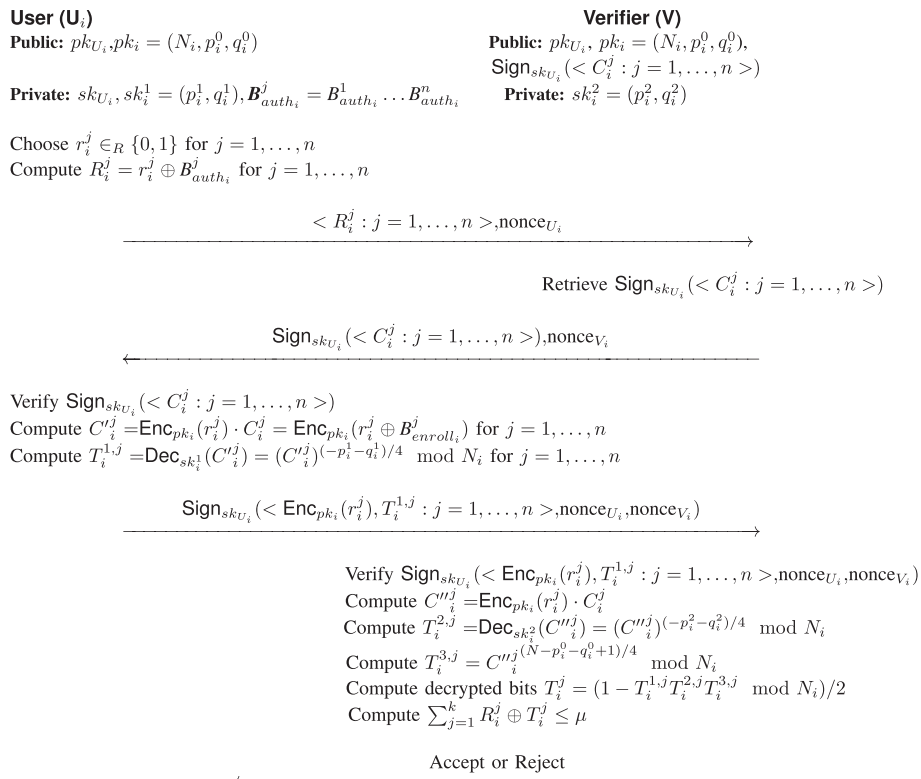
**Fig. 4** Authentication. Illustration of the THRIVE authentication stage



**Fig. 5** Enrollment protocol. The proposed enrollment protocol of the THRIVE system

$B_{auth_i}^1 \dots B_{auth_i}^n$  where  $B_{auth_i}^j \in \{0, 1\}, j = 1, \dots, n$ . Note that the user cannot produce exactly the same biometric template at each attempt and this results in different bihashes computed by the same user. Therefore,  $B_{enroll_i}$  and  $B_{auth_i}$  are different bihashes although they are generated by the same user at different sessions (e.g., enrollment and authentication stages). First,  $U_i$  chooses a random vector  $r_i^j \in_R \{0, 1\}$  for  $j = 1, \dots, n$ . She computes  $R_i^j = r_i^j \oplus B_{auth_i}^j$  for  $j = 1, \dots, n$ . Then,  $U_i$  generates a nonce,  $\text{nonce}_{U_i}$ , which is uniquely defined and contains information about user id, session id, and timestamp. Finally, the user sends  $< R_i^j : j = 1, \dots, n >, \text{nonce}_{U_i}$  to the verifier.

2. **Step 2:** The verifier retrieves  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$  from the database where  $C_i^j = \text{Enc}_{pk_i}(B_{enroll_i}^j)$  for  $j = 1, \dots, n$ . Then, it generates a nonce  $\text{nonce}_{V_i}$  which contains information about the verifier, session id, and timestamp. Finally, it sends  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >), \text{nonce}_{V_i}$  to the user.
3. **Step 3:** The user verifies  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$  by using public key  $pk_{U_i}$ . She computes  $C_i^j = \text{Enc}_{pk_i}(r_i^j) \cdot C_i^j = \text{Enc}_{pk_i}(r_i^j \oplus B_{enroll_i}^j)$  for  $j = 1, \dots, n$ . Then, she



**Fig. 6** Authentication protocol. The proposed authentication protocol of the THRIVE system

performs partial decryption over  $C_i^j$ , i.e.,  $T_i^{1j} = \text{Dec}_{sk_i^1}(C_i^j) = (C_i^j)^{(-p_i^1 - q_i^1)/4} \bmod N_i$ , for  $j = 1, \dots, n$  using her private key share  $sk_i^1$ . Finally, she sends  $\text{Sign}_{sk_{U_i}}(< \text{Enc}_{pk_i}(r_i^j), T_i^{1j} : j = 1, \dots, n >, \text{nonce}_{U_i}, \text{nonce}_{V_i})$  to the verifier.

**Step 4:**  $V$  verifies the signature

$\text{Sign}_{sk_{U_i}}(< \text{Enc}_{pk_i}(r_i^j), T_i^{1j} : j = 1, \dots, n >, \text{nonce}_{U_i}, \text{nonce}_{V_i})$  by using the public key  $pk_{U_i}$ . Then, it computes  $C_i^{mj} = \text{Enc}_{pk_i}(r_i^j) \cdot C_i^j$  (this is done to assure correctness of the result and will prevent a malicious user computing different values than expected.). Next, the verifier performs the full decryption by computing  $T_i^{2j} = \text{Dec}_{sk_i^2}(C_i^{mj}) = (C_i^{mj})^{(-p_i^2 - q_i^2)/4} \bmod N_i$  and  $T_i^{3j} = (C_i^{mj})^{(N - p_i^0 - q_i^0 + 1)/4} \bmod N_i$ . Finally, the verifier computes the decrypted  $j$ th bits  $T_i^j = (1 - T_i^{1j} T_i^{2j} T_i^{3j} \bmod N_i) / 2$  and the Hamming distance between  $R_i^j$  and  $T_i^j$  is calculated as follows

$$\sum_{j=1}^n R_i^j \oplus T_i^j \leq \mu, \quad (5)$$

where  $\mu$  is the distance threshold. Therefore, the verifier decides whether the user is authentic with respect to the pre-defined distance threshold. Note that the Hamming distance between  $r_i^j \oplus B_{enroll_i}^j$  and  $r_i^j \oplus B_{auth_i}^j$  is equal to the Hamming distance between  $B_{enroll_i}^j$  and  $B_{auth_i}^j$ . Finally, the verifier sends its decision (either Accept or Reject) to the user. However, the user may get dummy output if there is an error or an attack (i.e., override response attack) in the communication channel. The proposed system can easily be updated to cope with such an attack, for instance, by allowing the verifier to sign its decision including the nonces generated during the authentication session (i.e., either  $\text{Sign}(\text{Accept}, \text{nonce}_{U_i}, \text{nonce}_{V_i})$  or  $\text{Sign}(\text{Reject}, \text{nonce}_{U_i}, \text{nonce}_{V_i})$  and then sends it to the user. This way, authenticity, integrity, and origin of the data can easily be verified. Signing the nonces ( $\text{nonce}_{U_i}$  and  $\text{nonce}_{V_i}$ ) also makes the communication unique and avoids replay attacks.

**Lemma 3.** *Biohashes are not revealed at the authentication stage.*

*Proof.* Authentication is performed in a randomized domain. In other words, the authentication is determined

by comparing  $R_i^j$  and  $T_i^j$ . An adversary can only obtain  $R_i^j$  and  $T_i^j$  which are revealed at the authentication stage. Recall that these are randomized biohashes. Thus, from the adversary's perspective, there are three unknowns ( $r_i^j$ ,  $B_{enroll_i}^j$  and  $B_{auth_i}^j$ ) and two equations which are shown in the below.

$$T_i^j = r_i^j \oplus B_{enroll_i}^j \quad (6)$$

$$R_i^j = r_i^j \oplus B_{auth_i}^j \quad (7)$$

where  $r_i^j$  is the random bit generated by the  $U_i$  for the  $j$ th bit. Since this is a linear equation system with fewer equations than unknowns, it has many solutions. Consequently, it is impossible for the adversary to obtain a honest user's biohash by using  $T_i^j$  and  $R_i^j$  which are revealed at the authentication stage. As a result, the proposed biometric authentication system ensures security and privacy.  $\square$

## 5 Security proof of the proposed authentication protocol

In this section, we prove that the proposed authentication protocol shown in Fig. 6 is secure against malicious users and the verifier. More concretely, the primary goal of privacy preservation is to protect users' biometric templates during protocol executions. In a secure computation framework, parties have their own private input and are willing to evaluate a desired functionality  $f$  on their inputs without revealing any information except the outputs. This was originally formalized by Goldreich et al. in [80]. Intuitively, the following two scenarios should be absolutely indistinguishable: 1) securely computing  $f$  by realizing a protocol, and 2) privately sending their private inputs to a trusted third party, who then computes  $f$  and privately returns the outputs to each party. This formalization of secure computation is referred to as the simulation-based approach. The idea of the standard simulation-based privacy definition is that, given a well-defined privacy-leakage, a polynomial-time simulator (i.e., an adversary) can generate a transcript that is indistinguishable from the output of the real protocol. If such an efficient simulator exists, then an adversary cannot learn any additional information beyond the defined leakage. The simulator must perform its task without knowing the private information of the party who proves her identity [80].

In this proof, we show that given a party is corrupted (either user or verifier), there exists a simulator that can produce a view which is statistically indistinguishable

from the view of that party interacting with the other honest party. Assuming that one party is corrupted, we build an efficient simulator that has access to the public input and private secret shares of the secret key of the corrupted party. Besides, the simulator knows the public output. We want to point out that the simulator already knows the shares of the secret key of the corrupted party before the simulation is run. Since the threshold cryptosystem is set up before the protocol starts, we assume that the simulator extracts this information when the distributed key generation is run.

It is worth mentioning that the proposed authentication protocol gives computational privacy to both the user and the verifier due to the semantic security of the underlying cryptosystem. Furthermore, it is shown that the proposed authentication protocol is simulatable for both parties and these simulations produce views which are statistically indistinguishable from the views in the real protocol executions.

**Theorem 1.** *The proposed authentication protocol shown in Fig. 6 is secure in the presence of static malicious adversaries.*

*Proof.* We show that given a party is corrupted, there exists a simulator that can produce a view to the adversary that is statistically indistinguishable from the view in the real protocol execution based on its private decryption share as well as public information.  $\square$

*Case 1 - User  $U_i$  is corrupted.* In this case, we prove the security for the case where  $U_i$  is corrupted. The simulator has the private key share of the user  $sk_i^1$ , the user's private key  $sk_{U_i}$ , and the user's biohash  $B_{auth_i}^j$  apart from the user's public information (i.e.,  $pk_{U_i}$  and  $pk_i$ ) as described in the proposed authentication protocol. The simulator constructs a view for the user which is statistically close to the one the user observes when interacting with the honest verifier by using this information. The simulator proceeds as follows:

1. The simulator first obtains  $\langle R_i^j : j = 1, \dots, n \rangle, nonce_{U_i}$ . As in the second round of the real protocol, the simulator needs to output the signature of the encrypted biohash of the user. To do so, the simulator computes  $\tilde{C}_i^j = \text{Enc}_{pk_i}(B_{auth_i}^j)$  for  $j = 1, \dots, n$  by using the user's public key  $pk_i$ , and then computes  $\text{Sign}_{sk_{U_i}}(\langle \tilde{C}_i^j : j = 1, \dots, n \rangle)$ . The simulator also generates a nonce called  $nonce_{V_i}$ . The values  $\text{Sign}_{sk_{U_i}}(\langle \tilde{C}_i^j : j = 1, \dots, n \rangle)$  and  $nonce_{V_i}$  are the simulated outputs. Note that the simulator uses  $\tilde{\mathbf{B}}_{auth}$

instead of  $\mathbf{B}_{enroll}$  since it is the only available biohash to him.

2. The simulator obtains  $\text{Sign}_{sk_{U_i}}(\langle \text{Enc}_{pk_i}(r_i^j), T_i^{1,j} : j = 1, \dots, n \rangle, nonce_{U_i}, nonce_{V_i})$  as in the second round of the protocol. The simulator next verifies the signature  $\text{Sign}_{sk_{U_i}}(\langle \text{Enc}_{pk_i}(r_i^j), T_i^{1,j} : j = 1, \dots, n \rangle, nonce_{U_i}, nonce_{V_i})$  that  $U_i$  would run. Next, it computes  $\tilde{C}_i'^j = \text{Enc}_{pk_i}(r_i^j) \cdot \tilde{C}_i^j$ . Given  $\tilde{C}_i'^j$ , its plaintext and the share of private key  $sk_i^2$  of the user  $U_i$  the decryption shares  $T_i^{2,j}$  can be simulated as follows: The simulator computes  $\tilde{b}_0 = [\tilde{C}_i'^j]^{(N-p_0-q_0+1)/4} \bmod N$  from the public information and computes  $\tilde{b}_1 = [\tilde{C}_i'^j]^{(-p_1-q_1)/4} \bmod N$  since it knows  $sk_i^1$  (i.e.,  $p_1, q_1$ ). Let  $\tilde{b}$  denote the plaintext of  $\tilde{C}_i'^j$ . Then, the simulator can compute  $\tilde{b}_2 \bmod N \equiv (1 - 2\tilde{b})/(\tilde{b}_0\tilde{b}_1) \bmod N$  (which is  $T_i^{2,j}$  in the real protocol). Note that in the real setting this is not possible since the plaintext inside the ciphertext is unknown  $\tilde{C}_i'^j$ . Similarly,  $T_i^{3,j}$  can also be simulated since  $p_i^0$  and  $q_i^0$  are known by the simulator. The simulator finally computes  $\sum_{j=1}^k R_i^j \oplus T_i^j$ .

Each step of the proposed authentication protocol for the simulator is simulated and this completes the simulation for the malicious user. The transcript is consistent and statistically indistinguishable from the user's view when interacting with the honest verifier.

*Case 2 - The verifier  $V$  is corrupted.* We now prove the security for the case where the verifier is corrupted. The simulator has the private key share of the verifier ( $sk_i^2$ ) apart from the verifier's public information (i.e.,  $pk_{U_i}$ ,  $pk_i$ , and  $\text{Sign}_{sk_{U_i}}(\langle C_i^j : j = 1, \dots, n \rangle)$ ) as described in the proposed authentication protocol. The simulator constructs a view for the verifier which is statistically close to the one when interacting with the honest user by using this information. The simulator proceeds as follows:

1. Note that the simulator already knows  $\langle C_i^j : j = 1, \dots, n \rangle$  because of the knowledge of  $\text{Sign}_{sk_{U_i}}(\langle C_i^j : j = 1, \dots, n \rangle)$ . The simulator chooses a random bit  $\tilde{r}_i^j$  and arbitrary  $\tilde{B}_{auth_i}^j \in_R \{0, 1\}$  and computes  $\tilde{R}_i^j = \tilde{r}_i^j \oplus \tilde{B}_{auth_i}^j$ . Recall that the simulator must perform its task without knowing the private information of the honest user in this case. Thus, although it does not have real  $r_i^j$  and  $B_{auth_i}^j$ , it

can successfully execute the simulated conversation since  $\tilde{R}_i^j$  is uniformly random.

2. The simulator obtains  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$ . The simulator verifies the signature  $\text{Sign}_{sk_{U_i}}(< C_i^j : j = 1, \dots, n >)$  (by using the user's public key  $pk_{U_i}$ ) that  $V$  would run. Next, it next computes  $\tilde{C}_i^j = \text{Enc}_{pk_i}(r_i^j) \cdot C_i^j$ .
3. Given  $\tilde{C}_i^j$ , its plaintext (which is  $r_i^j \oplus B_{enroll_i}^j$ ) and the share of private key  $sk_i^{2j}$  of the verifier  $V$  the decryption share  $\tilde{T}_i^{1j}$  can also be simulated as follows:  
The simulator computes  $\tilde{b}_0 = [\tilde{C}_i^j]^{(N-p_0-q_0+1)/4} \bmod N$  from the public information and computes  $\tilde{b}_2 = [\tilde{C}_i^j]^{(-p_2-q_2)/4} \bmod N$  since it knows  $sk_i^2$  (i.e.,  $p_2, q_2$ ). Let  $\tilde{b}$  denote the plaintext of  $\tilde{C}_i^j$ . Then, the simulator can compute  $\tilde{b}_1 \bmod N \equiv (1 - 2\tilde{b}) / (\tilde{b}_0 \tilde{b}_2) \bmod N$  (which is  $T_i^{1j}$  in the real protocol). Note that in the real setting, this is not possible since the plaintext inside the ciphertext is unknown  $\tilde{C}_i^j$ .
4. Finally, the simulator needs to simulate the signature  $\text{Sign}_{sk_{U_i}}(< \text{Enc}_{pk_i}(r_i^j), T_i^{1j} : j = 1, \dots, n >, nonce_{U_i}, nonce_V)$ . However, this is not possible since the simulator does not know  $sk_{U_i}$ . In order to successfully simulate this final step, we need to provide additional information to the simulator. For example, performing a following modification over the key generation phase in the real protocol, the simulation will be possible:

- During the key generation in the real protocol, the private key  $sk_{U_i}$  is distributed to the user  $U_i$  and  $V$  in a threshold fashion. For example, distributed RSA setting can be used for the signature algorithm (note that for an RSA setting  $(e, n)$  denotes the public key and  $(p, q, d)$  denotes the private key where  $n = pq$  and  $ed \equiv 1 \bmod (p-1)(q-1)$ ). Namely, the private key  $d$  can be divided into  $d_1$  and  $d_2$  such that the ciphertext  $c$  can be decrypted together with  $U_i$  and  $V$  as  $m \equiv c^d \equiv c^{d_1+d_2} \bmod n$ .
- In order to simulate, instead of signing procedures,  $U_i$  will compute an encryption, compute its partial decryption and finally will send to the verifier. This will assure that the user indeed used its private decryption key over the encrypted value. Next, the verifier will also compute its partial decryption and will compute the decrypted value privately.

Thus, with this modified version the decryption share can be simulated in a similar way as described at the third step of the simulation.

Consequently, each step of the proposed authentication protocol for the simulator is simulated and this completes the simulation for the malicious verifier. The transcript is consistent and statistically indistinguishable from the verifier's view when interacting with the honest user.  $\square$

## 6 Complexity analysis of the proposed system

In this section, we discuss the complexity of the THRIVE enrollment and authentication protocols. The complexity of the THRIVE enrollment and authentication protocols are examined in terms of protocol steps for the round complexity, the number of cryptographic operations for the computational complexity and the number of messages exchanged by the two parties for the communication complexity. Without loss of generality, we will provide complexity of the THRIVE protocols using the (2,2)-threshold homomorphic GM cryptosystem as an instance [73]. In the protocol, we use (2,2)-threshold XOR-homomorphic GM cryptosystem for confidentiality (i.e., encryption and decryption) while for signature generation and verification a conventional cryptosystem such as RSA (using the key pair  $(pk_{U_i}, sk_{U_i})$ ) is employed.

The round complexity of the enrollment protocol is only one. For the computational complexity, the enrollment protocol requires  $n$  XOR-homomorphic encryptions, and one conventional signature generation for a user, but one signature verification for the server. For the communication complexity, the user sends a conventional signature and  $n$  ciphertexts (i.e.,  $C_i^j$  for  $j = 1, \dots, n$ ).

In the authentication protocol, there are only four rounds. For the computational complexity of the authentication protocol, the user generates one conventional signature and verifies another, computes  $n$  XOR-homomorphic encryptions and  $n$  XOR-homomorphic decryptions, and performs  $n$  modular multiplications over homomorphic ciphertexts (i.e.,  $\text{Enc}_{pk_i}(r_i^j) \cdot C_i^j$  for  $j = 1, \dots, n$ ). The verifier verifies one conventional signature, computes  $n + 2$  modular multiplications,  $2n$  decryptions, and performs  $n$  Jacobi computations to check  $\text{Enc}_{pk_i}(r_i^j)$  for  $j = 1, \dots, n$ . In total, there are  $n$  XOR-homomorphic encryptions,  $3n$  XOR-homomorphic decryptions, two signature verifications, one signature generation,  $n$  Jacobi computations, and  $2n + 2$  modular multiplications during the entire authentication protocol.

For the communication complexity of the authentication protocol, the user sends  $2n$  homomorphic ciphertexts, one conventional signature and one nonce value. The verifier sends one conventional signature,  $n$

homomorphic ciphertexts, and one nonce to the user. In total,  $3n$  homomorphic ciphertexts, two conventional signatures, and two nonce values are exchanged.

In the following, we provide timing estimates for the entire protocol for 80-bit security level, on a desktop computer, which has Intel processor with various clock speeds (2.4 and 3.2 GHz). On the computing platform independent from the clock speed, one modular multiplication using Montgomery arithmetic takes about 2000 clock cycles. Furthermore, one encryption operation in XOR-homomorphic GM cryptosystem takes only a single modular multiplication. On the other hand, one decryption in XOR-homomorphic GM cryptosystem requires one modular exponentiation operation which takes about 3.2 million clock cycles. In addition, one signature generation and verification operation in conventional cryptosystem such as RSA are equivalent to one modular exponentiation operation.

The bandwidth usage of the proposed protocol for various lengths of biohashes of the user are given in Table 1. The required bandwidth for the proposed protocol increases with the increasing length of the biohash. Bandwidth usage also affects the overall connection time.

The computation times for the user and the verifier at 2.4 GHz for the proposed protocol with different biohash lengths are given in Table 2. Naturally, it is expected that the required computation times of the proposed protocol increase as lengths of the biohashes increase.

In our timing estimates, we assume sequential (i.e., single-threaded) implementations of the user and server sides of the protocol that run on a single core. On the other hand, one advantage of the proposed system from the time complexity point of view is that majority of the expensive operations (i.e., mainly modular exponentiations) can be performed in parallel. Therefore, custom ASIC [81] or GPU implementations [82] can accelerate protocol considerably. For instance, the custom modular exponentiation circuit for RSA in [81] reports 0.89 ms execution time on a circuit of 153,000 equivalent gate counts. With specialized hardware that incorporates many custom modular exponentiation modules, overall execution time can be reduced significantly. Similarly, a GPU implementation is reported [82] to reach the peak throughput

**Table 2** Computation time for the user and the verifier at 2.4 GHz

Length of biohash	User time (ms)	Verifier time (ms)
112	151	449
192	258	769
256	343	1026
512	685	2050
2048	2735	8195

of 34,981 RSA-1024 decryptions per second with 2.6 ms latency. Apparently, a parallel GPU implementation of modular exponentiation will trivialize the computation complexity of the proposed technique. Finally, we expect an acceleration proportional to the number of cores if the protocol is implemented on a multi-core CPU platform.

We compare the communication complexity of the proposed system with the existing systems in the literature assuming that all systems run on a computer platform with 2.4 GHz clock speed. Erkin et al.'s system [41] requires 56.25 ms, Barni et al.'s system [20, 43] requires 50 ms and Sadeghi et al.'s system [42] requires 25 ms for authentication at the server side for single user with 112-bit binary feature vector [20]. On the other hand, our solution requires 449 ms for the same authentication setup. Although existing solutions seem faster than the proposed system, they propose their solutions in the semi-honest model whereas our solution is secure under malicious adversary model. The similar timing estimations are also computed for 3.2 GHz as shown in Table 3.

We also compare the communication complexity of the proposed system with those of the existing systems at 3.2 GHz in the literature. Kulkarni et al.'s system [64] requires 58 s at the server side, 10 ms at the user side, and 400 Kbit bandwidth usage for authentication of single user with 2048-bit binary feature vector at 3.2 GHz. Our proposed system requires 6146 ms at the server side, 2051 ms at the user side, and 6296 Kbit bandwidth usage for the same authentication setup. Thus, it is faster than Kulkarni et al.'s system. In addition, our proposed system offers other advantages such as that it is secure under malicious adversary model, that the biometric is protected via both a template protection method (e.g., biohash) and cryptographic

**Table 1** Bandwidth (total number of bits exchanged) usage of the proposed protocol

Length of biohash	Bandwidth (Kbits)	Time @ 10 Mbit/s (ms)
112	348	35
192	594	59
256	791	79
512	1577	158
2048	6296	630

**Table 3** Computation time for the user and the verifier at 3.2 GHz

Length of biohash	User time (ms)	Verifier time (ms)
112	113	337
192	193	577
256	257	769
512	514	1537
2048	2051	6146

primitives (e.g., threshold homomorphic encryption). On the other hand, Kulkarni et al.'s system offers solution for semi-honest model which also means that it is insecure for the malicious model. The comprehensive comparison between the proposed system and existing systems are shown in Fig. 7.

## 7 Comparison between the proposed system and the existing methods/systems

In this section, we compare the proposed THRIVE system with currently available biometric template protection solutions in the literature. The comprehensive comparison between the proposed system and existing solutions are shown in Fig. 7. This comparison is performed by checking whether they use or satisfy the following properties or not. These properties are given in the below but more information can be found in Sections 2 and 3.

- **Irreversibility:** It is computationally hard to reconstruct the original biometric template by using the protected biometric template (e.g., biohash, encrypted biometric data).
- **Renewability:** It is possible to generate different protected templates from the same biometric data of a user.
- **Diversity:** Different protected templates generated from the same biometric data of a user do not allow cross-matching or information leakage.
- **Helper data:** It is auxiliary data/side information, which is needed for running biometric system successfully (e.g., data for alignment, parity bits of error correction codes). Without use of helper data such biometric systems do not work properly. Furthermore, an attacker may deduce further information that can threat privacy of a user and security of a system from helper data.
- **Homomorphic encryption:** It is an encryption method, which allows computations to be carried out on ciphertext and obtaining the same results with the operations performed on the plaintext.
- **Threshold homomorphic encryption:** It is a special type of homomorphic encryption, which distributes the knowledge of a private key among  $n$  parties and at least  $t$  of the parties are required for successful decryption. This method can also ensure security of biometric data stored in the database. Even if biometric templates are encrypted, a database manager can easily decrypt them as soon as he knows the private key. However, threshold homomorphic encryption requires cooperation between parties for a successful decryption.
- **Semi-honest attack model:** In this adversary model, all parties follow the protocol but dishonest party may be curious to violate others privacy by keeping a record of all its intermediate computations and messages. This is weak attack model in comparison with

	Irreversibility	Renewability	Diversity	Helper Data	Homomorphic Encryption	Threshold Homomorphic Encryption	Semi-Honest Attack Model	Malicious Attack Model
<b>THRIVE System</b>	✓	✓	✓	✗	✓	✓	✗	✓
Salting Based Schemes	✗	✓	✓	✗	✗	✗	✗	✗
Non-Invertible Transform Based Schemes	✓	✓	✓	✗	✗	✗	✗	✗
Key Binding Schemes	✗	✓	✗	✓	✗	✗	✗	✗
Key Generation Schemes	✗	✗	✗	✓	✗	✗	✗	✗
Kerschbaum's System [62]	✓	✓	✓	✗	✓	✗	✓	✗
Erkin's System [42]	✓	✓	✓	✗	✓	✗	✓	✗
Barni's System [21,44]	✓	✓	✓	✗	✓	✗	✓	✗
Osadchy's System (SCIF) [65]	✓	✓	✓	✗	✓	✗	✓	✗
Rane's System [66]	✓	✓	✓	✗	✓	✗	✓	✗
Bringer's System (SHADE) [67]	✓	✓	✓	✗	✗	✗	✓	✗
Kulkarni's System [68]	✓	✓	✓	✗	✓	✗	✗	✗

**Fig. 7** Comparison. We compare the THRIVE system with the existing biometric template protection solutions with respect to various properties (e.g., irreversibility, renewability, diversity, helper data, homomorphic encryption, threshold homomorphic encryption, semi-honest attack model, malicious attack model). Check mark denotes that the system satisfies the property whereas X mark denotes that the system does not satisfy the property

malicious attack model. Semi-honest adversaries are also called honest-but-curious or passive attackers.

- **Malicious attack model:** In this adversary model, attacker makes arbitrary feasible deviation(s) from the protocol specification and does not have to follow instructions of the protocol. A malicious attacker can enter the protocol with arbitrary input, which may not be a true input, and this makes the malicious model harder to deal with. Malicious adversaries are also called active attackers.

It is clearly seen in Fig. 7 that the proposed THRIVE system offers superior security and privacy preservation solutions under the malicious attack model.

## 8 Conclusion

In this work, we propose a novel biometric authentication system. The aim of the THRIVE system is to increase security against adversary attacks defined in [83] and preserve the privacy of users. The proposed system can be used with any biometric feature extraction method which can produce binary templates or whose templates can be binarized by post-processing. The biohashing is chosen as an example binary biometric template generation system since it offers satisfactory performance and fast authentication. The comparison is performed in a randomized domain at the authentication stage and the binary templates (e.g., biohashes) are never released. In addition, only encrypted binary templates are stored in the database. Since we use the (2, 2)-threshold cryptosystem, the verifier cannot decrypt the data stored in the database by itself. Namely, the user and the verifier both has to cooperate to decrypt the encrypted binary templates. The THRIVE system can be used in applications where the user is not willing to reveal her biometrics to the verifier although she needs to proof her physical presence by using biometrics. It is also suitable for applications where the user and the verifier do not necessarily trust each other. The THRIVE system appears to be sufficiently efficient compared to the existing scheme and can be used in real-life applications. A common drawback of all existing schemes (including the THRIVE system), which consider the existence of malicious verifiers, is to utilize expensive asymmetric encryptions. Providing more efficient constructions (i.e., less communication, storage and computational overheads) while ensuring user privacy in the presence of malicious verifiers is still an open problem.

Biometric authentication systems work with some error rates which can be represented by using equal error rate (EER), false acceptance rate (FAR), and false rejection rate (FRR). These error rates may occur due to the nature of biometric data and can vary with respect to the various factors. The proposed THRIVE system can work with any biometric authentication scheme whose outputs can

be binarized. The main goal of the THRIVE system is to increase security of authentication process and enhance privacy of users. Error rates (e.g., EER, FAR, FRR) of the THRIVE system related with authentication process is dependent on chosen biometric modality, underlying feature extraction method, and biometric authentication scheme (e.g., biohashing is given as an example in this paper).

## Endnote

<sup>1</sup>A nonce is an arbitrary number used only once in a cryptosystem.

## Competing interests

The authors declare that they have no competing interests.

## Acknowledgements

This work has been supported by the BEAT project 7<sup>th</sup> Framework Research Programme of the European Union (EU), grant agreement number: 284989. The authors would like to thank the EU for the financial support and the partners within the consortium for a fruitful collaboration. For more information about the BEAT consortium, please visit <http://www.beat-eu.org>.

## Author details

<sup>1</sup>TÜBİTAK BİLGEM UEKAE, Gebze, Kocaeli, Turkey. <sup>2</sup>Sabancı University, Tuzla, Istanbul, Turkey.

Received: 23 February 2015 Accepted: 21 July 2015

Published online: 07 August 2015

## References

1. AK Jain, A Ross, S Prabhakar, An introduction to biometric recognition. *IEEE Trans. Circuits Syst. Video Technol.* **14**, 4–20 (2004)
2. S Prabhakar, S Pankanti, AK Jain, Biometric recognition: Security and privacy concerns. *IEEE Secur. Privacy.* **1**(2), 33–42 (2003)
3. AK Jain, K Nandakumar, A Nagar, Biometric template security. *EURASIP J. Adv. Signal Process.* **2008**, 113–11317 (2008)
4. NK Ratha, JH Connell, RM Bolle, Enhancing security and privacy in biometrics-based authentication systems. *IBM Syst. J.* **40**(3), 614–634 (2001)
5. C Roberts, Biometric attack vectors and defences. *Comput. Secur.* **26**(1), 14–25 (2007)
6. F Hao, R Anderson, J Daugman, Combining crypto with biometrics effectively. *IEEE Trans. Comput.* **55**(9), 1081–1088 (2006)
7. S Kanade, D Petrovska-Delacretaz, B Dorizzi, in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference On*. Cancelable iris biometrics and using error correcting codes to reduce variability in biometric data (Miami, FL, 2009), pp. 120–127
8. A Juels, M Wattenberg, in *CCS '99, Proceedings of the 6th ACM Conference on Computer and Communications Security*. A fuzzy commitment scheme (Singapore, 1999), pp. 28–36
9. TAM Kevenaar, GJ Schrijen, M van der Veen, AHM Akkermans, F Zuo, in *Proceedings of the Fourth IEEE Workshop on Automatic Identification Advanced Technologies*. AUTOID '05. Face recognition with renewable and privacy preserving binary templates, (2005), pp. 21–26. doi:10.1109/AUTOID.2005.24
10. GI Davida, Y Frankel, BJ Matt, in *IEEE Symposium on Security and Privacy*. On enabling secure applications through off-line biometric identification (Oakland, CA, 1998), pp. 148–157
11. S Tulyakov, F Farooq, V Govindaraju, in *ICAPR (2)*. Symmetric hash functions for fingerprint minutiae, (2005), pp. 30–38. doi:10.1007/11552499\_4
12. Y Sutcu, Q Li, N Memon, in *SPIE Conf on Security, Steganography and Watermarking of Multimedia Contents IX*. How to protect biometric templates, (2007). doi:10.1007/978-3-319-07857-1\_27
13. A Stoianov, T Kevenaar, M van der Veen, in *Science and Technology for Humanity (TIC-STH), 2009 IEEE Toronto International Conference*. Security issues of biometric encryption (Toronto, ON, 2009), pp. 34–39



14. X Zhou, Privacy and security assessment of biometric template protection. *IT - Inform. Technol.* **54**(4), 197 (2012)
15. X Zhou, A Kuijper, R Veldhuis, C Busch, in *Proceedings of the 2011 International Joint Conference on Biometrics*, IJCB '11. Quantifying privacy and security of biometric fuzzy commitment (IEEE Computer Society Washington, DC, USA, 2011), pp. 1–8
16. S Cimito, M Gamassi, V Piuri, R Sassi, F Scotti, in *CIS. A biometric verification system addressing privacy concerns* (Harbin, 2007), pp. 594–598
17. T Matsumoto, H Matsumoto, K Yamada, S Hoshino, Impact of artificial "gummy" fingers on fingerprint systems. *Datenschutz und Datensicherheit*. **26**(8) (2002). doi:10.1117/12.462719
18. T van der Putte, J Keuning, in *CARDIS. Biometrical fingerprint recognition: Don't get your fingers burned*, (2000), pp. 289–306. ISBN:0-7923-7953-5
19. J Bringer, H Chabanne, GD Cohen, B Kindarji, G Zémor, Optimal iris fuzzy sketches. *CoRR abs/0705.3740* (2007). doi:10.1109/BTAS.2007.4401904
20. M Barni, T Bianchi, D Catalano, M Di Raimondo, R Donida Labati, P Failla, D Fiore, R Lazzeretti, V Piuri, F Scotti, A Piva, in *Proceedings of the 12th ACM Workshop on Multimedia and Security. Privacy-preserving fingerprint authentication* (ACM New York, NY, USA, 2010), pp. 231–240
21. K Nandakumar, AK Jain, S Pankanti, Fingerprint-based fuzzy vault: Implementation and performance. *IEEE Trans. Inform. Forensics Secur.* **2**(4), 744–757 (2007)
22. YC Feng, PC Yuen, AK Jain, A hybrid approach for generating secure and discriminating face template. *Trans. Info. For. Sec.* **5**(1), 103–117 (2010)
23. FM Bui, K Martin, H Lu, KN Plataniotis, D Hatzinakos, Fuzzy key binding strategies based on quantization index modulation (QIM) for biometric encryption (be) applications. *IEEE Trans. Inform. Forensics Secur.* **5**(1), 118–132 (2010)
24. A Juels, M Sudan, A fuzzy vault scheme. *Des. Codes Cryptography*. **38**(2), 237–257 (2006)
25. C Karabat, H Erdogan, in *Proceedings of the 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing. A cancelable biometric hashing for secure biometric verification system* (Kyoto, 2009), pp. 1082–1085
26. Z Bai, D Hatzinakos, in *Control Automation Robotics Vision (ICARCV), 2010 11th International Conference On. Lbp-based biometric hashing scheme for human authentication* (Singapore, 2010), pp. 1842–1847
27. YW Kuan, ABJ Teoh, DCL Ngo, Secure hashing of dynamic hand signatures using wavelet-fourier compression with biophasor mixing and 2n discretization. *EURASIP J. Adv. Sig. Proc.* **2007** (2007). doi:10.1155/2007/59125
28. C Rathgeb, A Uhl, Iris-biometric hash generation for biometric database indexing. *Pattern Recognit. Int. Conf.*, 2848–2851 (2010). doi:10.1109/ICPR.2010.698
29. R Lumini, L Nanni, An improved bihashing for human authentication. *Pattern Recognit.* **40**, 1057–1065 (2006)
30. WJ Scheirer, TE Boulton, in *Proceedings of Biometrics Symposium. Cracking fuzzy vaults and biometric encryption* (Baltimore, MD, 2007), pp. 1–6
31. A Adler, in *In International Conference on Audio and Video Based Biometric Person Authentication. Vulnerabilities in biometric encryption systems* (Springer-Verlag Berlin, Heidelberg, 2005), pp. 1100–1109
32. TE Boulton, WJ Scheirer, R Woodworth, in *CVPR. Revocable fingerprint biotokens: Accuracy and security analysis* (Minneapolis, MN, 2007)
33. A Kong, K-H Cheung, D Zhang, M Kamel, J You, An analysis of bihashing and its variants. *Pattern Recogn.* **39**, 1359–1368 (2006)
34. K Kommel, C Vielhauer, in *Proceedings of the 12th ACM Workshop on Multimedia and Security. MMSec '10. Reverse-engineer methods on a biometric hash algorithm for dynamic handwriting* (ACM New York, NY, USA, 2010), pp. 67–72
35. KH Cheung, AW-K Kong, J You, D Zhang, in *CISST. An analysis on inevitability of cancelable biometrics based on bihashing* (Springer-Verlag Berlin, Heidelberg, 2005), pp. 40–45
36. K Kummel, C Vielhauer, T Scheidat, D Franke, J Dittmann, in *Proceedings of the 11th IFIP TC 6/TC 11 International Conference on Communications and Multimedia Security. Handwriting biometric hash attack: a genetic algorithm with user interaction for raw data reconstruction*, (2010), pp. 178–190. doi:10.1007/978-3-642-13241-4\_17
37. K Simoens, P Tuyts, B Preneel, in *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy. IEEE SP'09. Privacy weaknesses in biometric sketches* (Berkeley, CA, 2009), pp. 188–203
38. T Ignatenko, FMJ Willems, Information leakage in fuzzy commitment schemes. *IEEE Trans. Inform. Forensics Security*. **5**(2), 337–348 (2010)
39. K Simoens, J Bringer, H Chabanne, S Seys, A framework for analyzing template security and privacy in biometric authentication systems. *IEEE Trans. Inform. Forensics Secur.* **7**(2), 833–841 (2012)
40. T Ignatenko, FMJ Willems, Biometric systems: privacy and secrecy aspects. *Trans. Info. For. Sec.* **4**(4), 956–973 (2009)
41. Z Erkin, M Franz, J Guajardo, S Katzenbeisser, I Legendijk, T Toft, in *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies. PETS '09. Privacy-preserving face recognition* (Seattle, WA, USA, 2009), pp. 235–253
42. A-R Sadeghi, T Schneider, I Wehrenberg, in *ICISC. Efficient privacy-preserving face recognition* (Springer-Verlag Berlin, Heidelberg, 2009), pp. 229–244
43. M Barni, T Bianchi, D Catalano, M Di Raimondo, RD Labati, P Failla, D Fiore, R Lazzeretti, V Piuri, A Piva, F Scotti, in *Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference On. A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingerprint templates* (Washington, DC, Sept. 2010)
44. Y Sutcu, HT Sencar, N Memon, in *Proceedings of the 7th Workshop on Multimedia and Security. A secure biometric authentication scheme based on robust hashing* (ACM New York, NY, USA, 2005), pp. 111–116
45. ATB Jin, K-A Toh, WK Yip, in *ICB. 2<sup>nd</sup> discretisation of biophasor in cancellable biometrics* (Seoul, Korea, 2007), pp. 435–444
46. B Yang, C Busch, P Bours, D Gafurov, in *Media Forensics and Security. Robust minutiae hash for fingerprint template protection* (San Jose, CA, USA, 2010)
47. U Uludag, S Pankanti, AK Jain, in *Proceedings of the IEEE. Biometric cryptosystems: Issues and challenges*, (2004), pp. 948–960. doi:10.1109/JPROC.2004.827372
48. C Rathgeb, A Uhl, in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference On. Statistical attack against iris-biometric fuzzy commitment schemes* (Colorado Springs, CO, 2011), pp. 23–30
49. T Ignatenko, F Willems, in *Proceedings of the 2009 IEEE International Conference on Symposium on Information Theory - Volume 4. ISIT'09. Secret rate - privacy leakage in biometric systems* (Seoul, 2009), pp. 2251–2255
50. A Stoianov, in *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference On. Security of error correcting code for biometric encryption* (Ottawa, ON, 2010), pp. 231–235
51. E-C Chang, R Shen, FW Teo, in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security. Finding the original point set hidden among chaff* (ACM New York, NY, USA, 2006), pp. 182–188
52. A Kholmatov, B Yanikoglu, Realization of correlation attack against the fuzzy vault scheme (2008). doi:10.1117/12.766861
53. Y Dodis, R Ostrovsky, L Reyzin, A Smith, Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.* **38**(1), 97–139 (2008)
54. Y Dodis, B Kanukurthi, J Katz, L Reyzin, A Smith, Robust fuzzy extractors and authenticated key agreement from close secrets. *IEEE Trans. Inform. Theory*. **58**(9), 6207–6222 (2012)
55. Y Sutcu, L Qiming, N Memon, in *Proceedings of SPIE Optics and Photonics in Global Homeland Security V and Biometric Technology for Human Identification VI. Design and analysis of fuzzy extractors for faces*, vol. 7305, (2009). doi:10.1117/12.820571
56. TS Ong, ABJ Teoh, in *Proceedings of the Third International Symposium on Information Assurance and Security. Fuzzy key extraction from fingerprint biometrics based on dynamic quantization mechanism* (Manchester, 2007), pp. 71–76
57. A Arakala, J Jeffers, KJ Horadam, in *ICB'07. Fuzzy extractors for minutiae-based fingerprint authentication* (Seoul, Korea, 2007), pp. 760–769
58. P Tuyts, AHM Akkermans, TAM Kevenaar, GJ Schrijen, AM Bazen, RNJ Veldhuis, in *AVBPA. Practical biometric authentication with template protection*, (2005), pp. 436–446. doi:10.1007/11527923\_45
59. X Boyen, in *Proceedings of the 11th ACM Conference on Computer and Communications Security. CCS '04. Reusable cryptographic fuzzy extractors*, (2004), pp. 82–91. doi:10.1145/1030083.1030096

60. MG Qiming Li, E-C Chang, in *Proceedings of IEEE Workshop on Biometrics (In Association with CVPR)*. Fuzzy extractors for asymmetric biometric representation (Anchorage, AK, 2008)
61. F Kerschbaum, MJ Atallah, D M'Raihi, JR Rice, in *ICBA*. Private fingerprint verification without local storage (Hong Kong, China, 2004), pp. 387–394
62. RW Hamming, Error detecting and error correcting codes. *Bell Syst. Technical J.* **29**, 147–160 (1950)
63. MA Turk, AP Pentland, in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Face recognition using eigenfaces (Maui, HI, 1991), pp. 586–591
64. R Kulkarni, A Namboodiri, in *Biometrics (ICB), 2013 International Conference On*. Secure hamming distance based biometric authentication (Madrid, 2013), pp. 1–6
65. D Boneh, E-J Goh, K Nissim, in *Proceedings of the Second International Conference on Theory of Cryptography*. TCC'05. Evaluating 2-dnf formulas on ciphertexts (Springer Berlin, Heidelberg, 2005), pp. 325–341
66. M Osadchy, B Pinkas, A Jarrous, B Moskovich, in *Security and Privacy (SP), 2010 IEEE Symposium On*. Scifi - a system for secure face identification (Oakland, CA, USA, 2010), pp. 239–254
67. SD Rane, W Sun, A Vetro, in *Image Processing (ICIP), 2009 16th IEEE International Conference On*. Secure distortion computation among untrusting parties using homomorphic encryption (Cairo, 2009), pp. 1485–1488
68. J Bringer, H Chabanne, A Patey, in *Financial Cryptography Workshops*. Shade: Secure hamming distance computation from oblivious transfer (Okinawa, Japan, 2013), pp. 164–176
69. MS Kiraz, B Schoenmakers, J Villegas, in *Information Security, 10th International Conference, ISC 2007, Valparaíso, Chile, October 9-12, 2007, Proceedings*. Efficient committed oblivious transfer of bit strings, (2007), pp. 130–144
70. T El Gamal, in *Proceedings of CRYPTO 84 on Advances in Cryptology*. A public key cryptosystem and a signature scheme based on discrete logarithms, (1985), pp. 10–18. doi:10.1109/TIT.1985.1057074
71. P Paillier, D Pointcheval, in *Advances in Cryptology - Proceedings of ASIACRYPT '99*. LNCS, ed. by K-Y Lam, E Okamoto, and C Xing. Efficient public-key cryptosystems provably secure against active adversaries, vol. 1716 (Springer Singapore, 1999), pp. 165–179
72. J Katz, M Yung, Threshold cryptosystems based on factoring. **2501**, 192–205 (2002). doi:10.1007/3-540-36178-2\_12
73. S Goldwasser, S Micali, Probabilistic encryption. *J. Comput. Syst. Sci.* **28**(2), 270–299 (1984)
74. R Cramer, I Damgård, JB Nielsen, in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology*. EUROCRYPT '01. Multiparty computation from threshold homomorphic encryption, (2001), pp. 280–299. doi:10.1007/3-540-44987-6\_18
75. M Raginsky, S Lazebnik, in *Advances in Neural Information Processing Systems*. Locality-sensitive binary codes from shift-invariant kernels, (2009), pp. 1509–1517. web.engr.illinois.edu
76. A Gionis, P Indyk, R Motwani, in *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999*. Similarity search in high dimensions via hashing (Edinburgh, Scotland, UK, 1999), pp. 518–529
77. J Bringer, H Chabanne, B Kindarji, Identification with encrypted biometric data. *Secur. Commun. Netw.* **4**(5), 548–562 (2011)
78. MS Charikar, in *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*. STOC '02. Similarity estimation techniques from rounding algorithms (ACM New York, NY, USA, 2002), pp. 380–388
79. DCL Ngo, ATB Jin, A Goh, Biometric hash: high-confidence face recognition. *IEEE Trans. Circuits Syst. Video Techn.* **16**(6), 771–775 (2006)
80. O Goldreich, *Foundations of Cryptography: Volume 1*. (Cambridge University Press, New York, NY, USA, 2006)
81. A Miyamoto, N Homma, T Aoki, A Satoh, Systematic design of RSA processors based on high-radix montgomery multipliers. *IEEE Trans. VLSI Syst.* **19**(7), 1136–1146 (2011)
82. Y Yang, Z Guan, H Sun, Z Chen, in *Information Security Practice and Experience*. Lecture Notes in Computer Science, ed. by J Lopez, Y Wum. Accelerating RSA with fine-grained parallelism using GPU, vol. 9065 (Beijing, China, 2015), pp. 454–468
83. NK Ratha, JH Connell, RM Bolle, in *Proc. 3rd AVBPA*. An analysis of minutiae matching strength (Halmstad, Sweden, 2001), pp. 223–228

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- Convenient online submission
- Rigorous peer review
- Immediate publication on acceptance
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

---

Submit your next manuscript at ► [springeropen.com](http://springeropen.com)